4:th CEAS Air & Space Conference

FTF Congress: Flygteknik 2013

# Advantages of an Integrated Simulation Environment

**Markus Kunde and Andreas Schreiber**
*Simulation and Software Technology*
*German Aerospace Center (DLR)*
*51147 Cologne, Germany*

**Keywords:** Simulation, MDAO, Integration, CPACS, RCE

**Abstract**

Numerical simulation software in the field of aircraft design can be classified as first, second or third generation Multi-Disciplinary Analysis and Optimatzion (MDAO) system. The most challenging task nowadays is to create a third generation MDAO because there are no good-practice rules how to create a useful software system. The prerequisite for a success-story of such a software is a successful consideration of the dependency of simulation scenario (workflow), simulation models and the simulation data. Some good-practice rules for developing a third generation MDAO can be extracted from monolithic first generation systems regarding this dependency. The still under development software *RCE for CPACS* is a system for applied numerical aviation pre-design simulations and a technology carrier for evolving a third generation MDAO. The current state of *RCE for CPACS* regarding the good-practice rules will be outlined. A future state will be sketched considering possible next steps on the way to a successful third generation MDAO.

## 1 Introduction

Simulation in the aircraft engineering domain is a challenging task. Different knowledge domains have to work together to fulfill the requirements each domain has regarding the aircraft. From a historical view there are three generations of systems which meet this challenging task [1]. First generation Multi-Disciplinary Analysis and Optimization (MDAO) systems are described as single user operating a monolithic simulation workflow. The second generation is described as a distributed system with different simulation models but with a single user. The third generation consist of a distributed system with multiple users, representing design or model experts.
In this paper, we describe the third generation MDAO system of the German Aerospace Center (DLR). It is developed, maintained and used by most aeronautic institutes of DLR, especially the institutes for Aerodynamics and Flow Technology, Air Transportation Systems, and Simulation and Software Technology.

## 2 Advantages of an Integrated Simulation Environment

Third generation MDAO systems are still under development and therefore there is no best practice answer how such a system should be designed. In the following we will focus on third generation MDAO systems as this research is done in the field of distributed simulation systems. It may be possible to transfer the results to monolithic systems as well, but usually their internal structure follows the concept we want to achieve in a distributed system. Within third generation MDAO systems there is a collaborative point of view onto multidisciplinary scenarios which include the relationship and dependency of each domain to the others. Ques-

Table 1: Structure of simulation aspects

| **Aspect** | **Technical View** |
|---|---|
| Workflow | Simulation environment for execution of simulation workflows. Software which is able to perform simulations. Needs integration possibility of simulation models. Should be able to combine models and describe their relationship. |
| Model | Algorithm which represents domain specific knowledge. Possibility to insert data, calculate algorithm and receiving a result. |
| Data | Data item which describes the knowledge domain(s). |

tions like "how should a simulation perform?" or "what supporting features are needed to build up a simulation scenario?" should be answered [2]. But there is also a technical view on these simulation scenarios possible. Each simulation consists of three aspects:

**Simulation Workflow.** A simulation workflow represents the complete simulation scenario and consists of simulation models and their relations.

**Simulation Models.** A simulation model describes a self-containing (coherence) simulation aspect. Combining two or more models result in a simulation workflow description.

**Simulation Data.** The data representing the result of each simulation model and therefore the result of the simulation workflow.

This collection represents the lowest common denominator for any simulation and without one of them a simulation scenario is not possible. Having a closer technical look at each aspect shows the usual structure (see Table 1).

It is obvious that each simulation aspect has a relationship to another (see Figure 1). The typical relationship observed in many distributed simulation environments is: "Simulation workflow knows simulation model which knows simulation data". This approach has some disadvantages because no aspect has knowledge about the others. The workflow has no knowledge about the current content of the simulation data and the complete simulation cannot react on it. The simulation model does not have any knowledge about the workflow system and cannot act well in the workflow context. In the following we will describe our approach with its advantages to tackle this disharmonic situation. These benefits touch the observed drawbacks in the typical relationship mentioned above.

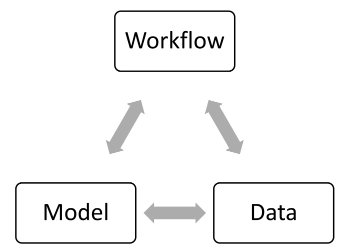


Figure 1: Dependency cycle of simulation aspects.

## 2.1 Structural View on MDAO systems

In a third generation MDAO system where workflow, model and data are integrated into each other several advantages exist. The structure of such a solution will be described in the following. A concrete solution will be introduced in section 3.3.1. From an engineers point of view the integrated solution looks as follows:

**Simulation Workflow.** The simulation workflow system is a software where simulation models and data are integrated. The system considers at least a concept how to integrate a model with its specifics, e.g., their data connections or computer environment. It shows specific views onto the model and let the user make use of the model in its native or in a predefined conceptual way. The data format is considered in the system as well. Specific possibilities for choosing data items out of the data or specific views are necessary. In the highest integration dimension data format specifics are hidden and an aircraft engineers view onto the data appears.

**Simulation Model.** The simulation model is aware about the data format. It integrates in a naturally way into the internal algorithm of the model. There is no conversion to own data format(s) necessary regarding any loss of information during conversion. The model considers the simulation workflow system. It uses its advantages and its way to integrate a model into the workflow system.

**Simulation Data.** The simulation data consist of all data artefacts all models need. It describes the full dimension of the simulation task.

**Supporting Simulation Software Libraries.** Supporting libraries are a very common way to centralize parts of the simulation algorithms. The access to the data format or basic calculations can be made at this point and simulation models and workflow system can make use of it.

The following set of rules describe the advantages of an integrated MDAO system. They were extracted from the described structure above and evolved to our approach.

**Hiding data format specifics in simulation workflow system.** Hiding data specifics in a workflow system under a more engineers related view implies a flat learning curve instead of a high initial effort. Persons who are new to this research topic are affected as well as every involved person when, e.g., a change in the data format is made.

**Way of working fits to data format, models and workflow.** Every workflow system, every simulation model and even a data format have their own way how they work and how they should be used. When all aspects have the same view on data, model and workflow, they act in the same way and do not contradict in their processing. This is the case when a model-driven approach meets a data- oder process-driven one. These different approaches fit rarely together.

**Affecting changes in data format usually results in changes of simulation environment.** The direct dependency from data format to model and workflow system results in a direct change of both when something meaningful is changed in the data format. This is in most situations a good approach for the user because he always has the most up-to-date information about the data format. There is of course also a direct dependency between workflow system and model. These dependencies have the manner to be loosely coupled and therefore have a continous modification character instead of being too much or to less regarding their frequency and substantial.

**Same calculation basis.** The same calculation basis (regarding the supporting software libraries) have a very high effect to the quality of the results. When using a common shared software library there is no discrepancy between the simulation models with respect to their internal assumptions about the data. The same access to the data format can be used as well as some shared calculation algorithms. This affects a single point of failure regarding results.

**Working direct on data.** As each aspect, workflow system and simulation models, have direct access to the data format they can make use of it as their internal data model. This has the benefit that no conversion irreversibilities can occur. This sort of error has the same high effect on the quality of calculation results like the same calculation basis.

**React on data internals.** Then the workflow system is aware of the data format and is able to comprehend data changes it is possible to create a more intelligent system. It can prevent the user from mistakes or may react on changes regarding the workflow.

### 2.2 Generic Proposal of MDAO System

In a more concrete detailed view, a typical MDAO system should not regard only some rules of how workflow, models and data fit together. It should consider also different kind of insights how such a system should act with respect to some software usability guidelines. Therefore a workflow containing models which interact with data is only one view onto a simulation. Depending on the fact that there are several different disciplines with different perspectives onto a simulation or a part of a simulation, each of these different groups should be considered with a special view on a simulation. During designing an MDAO system a clear identification of every user group and its look onto the simulation should be made. Table 2 shows an extract of some possible user groups.

In general each discipline should work on their

Table 2: Extract of possible user groups.

| User Group | View Description |
|---|---|
| Engine Designer | Should see the engine in detail. Engine parts or fluids flow are in focus of interest. |
| Aero-dynamics Designer | Interested in a detailed geometry view, e.g., with some dynamics. Possibility to modify geometry should be given. |
| Economics Designer | Valueing the aircraft or fleet with some boundary conditions. View on data is a strong calculation and formula one. |
| Aero-elastics Designer | Has a similar view on geometry like the aerodynamics designer, but with some other focus. |
| Simulation Designer | Is interested in the complete simulation workflow. Should see the dependencies between different disciplines. |
| Data Quality Analyst | The consistency and correctness of the data are in main focus. The main question is if the simulation outcome is valid and trustable. |

own topics with their own 'natural' view onto their special part of the simulation. This makes it more easy to create a simulation and to evaluate the results.

The separation into views and the underlying simulation structure has a strong decoupling effect which is neccessary for avoiding too heavy maintenance effort. The simulation aspects itself (workflow, models and data) should be strongly coupled together as there is a strong dependency to each other.

### 2.3 Benefiting Effects of Coupling

As already described in section 2.1 the coupling of workflow, models and data has some benefiting effects we will focus more concrete in the following. Therefore a closer look at the roles of each aspect should be made.

**Simulation Workflow.** The simulation workflow has some more tasks than connecting simulation models and the regulation of data flow. Following the main topics a workflow should be aware of:

- Knows dependencies between models or models and data
- Prevents user from usage-errors
- Assists of design task
- Knows how a model works (for control) and for what tasks it can be used
- Gives statement to quality of results
- Represents data in adequate way
- Reacts in a smart (intelligent) way on any changes (with, e.g., modified process execution)

All of these points ensure the outcome of the simulation regarding consistency and correctness of data. In other words, the simulation workflow is responsible for trustability of the result.

**Simulation Models.** The simulation model represents a specific aviation discipline. Nethertheless it needs knowledge of the outside environment. Following the main topics a model should be aware of:

- Knows how workflow works (model-, data- or process-driven) and support this behavior
- Ensures consistency of data regarding all data which are adressed by the model

These points affect the model and its outcome. A model is thereby always responsible for the quality of the result.

**Simulation Data.** The simulation data represents at least the complete simulation data-artefacts. But it has some more tasks as being a container for aviation-results:

- Knows which data items influence each other

- Knows which models access to specific data items and therefore data knows the dependencies between the models
- Knows every change of a data item and track the changes each model performs

Summarizing the aspects of the workflow (responsible of trustability of the result), models (responsible of quality of the result) and data (the outcome of the simulation) are equally important to a simulation environment. As all of them have dependencies to the outcome it is necessary that every aspect is aware of the other two aspects.

### 2.4 Example "VAMPzero"

The advantages can best shown on a monolithic system because the concept described above takes the advantages of a monolithic system to a third generation MDAO system with its distributed nature. The reference example is the simulation model "VAMPzero" [3]. VAMPzero is a conceptual aircraft design tool based on handbook methods. It integrates the CPACS data format [4] and is model and workflow system in one tool. VAMPzero is able to trace changes on data because it directly tracks relationships between the used algorithm and the underlying data item. Therefore it can resolve dependency forward ("what data item/algorithm is based on the current calculated algorithm") and backward ("from what data item/algorithm is the current calculated algorithm addicted"). There is a full recognition of what is happening during calculation and therefore the trust and quality of the outcome is extremly high. Changes in the workflow or model or data aspect is directly available to the other aspects as well because they are related to each other. We transfered this integrated concept to a third generation MDAO system.

## 3 Integrated third generation MDAO system

The integrated third generation MDAO system of German Aerospace Center (DLR) consist of all three simulation aspects and one additional aspect. The simulation aspects are identified in 2.4 while the additional aspect (supporting software libaries) is introduced in the same section. To come to the concrete simulation environment at DLR, a mapping to concrete names of software should be introduced (Table 3).

Table 3: Simulation aspects at DLR

| **Aspect** | **Concrete DLR Software** |
|---|---|
| Workflow | RCE for CPACS |
| Model | Several simulation algorithms which implement the "CPACS-"way of integration into RCE |
| Data | CPACS data format |
| Libraries | TiXI & TiGL |

The section will describe each aspect on its own.

### 3.1 Data Format - CPACS

The Common Parametric Aircraft Configuration Scheme (CPACS) [4] is

> "one potential step towards a unified data model".

B. Nagel et al. introduce CPACS as the data format which is designed for performing collaborative MDAO simulations in aircraft design. The format of CPACS is human readable and provides a computer a structured access. It is based on Extensible Markup Language (XML) and is defined as a schema definition (XSD). CPACS has a hierarchical structure based on the parts of an aircraft and can describe, beside the entire vehicle, many related aspects of aircraft design (e.g., missions or fleets). The format handles product and process information. Each part of the aircraft is called component. In the hierarchical structure assemblies can be defined which are based on components. They can be used multiple times via a referencing approach with Unique Information Identifiers (UIDs). This avoids redundancies in the data format. On top of W3C XML specification [1] two techniques are added. At first external data-files can be referenced in the XML data format and will included

[1] http://www.w3.org/XML, 01.03.2013

during reference into a sub-node of the CPACS data. The second addition is CPACS-support for multi-dimension arrays which can be specified. More information about CPACS can be found in [5, 6].

### 3.2 Supporting Libraries - TiXI & TiGL

There are two supporting libraries based on CPACS data format.

**TiXI.** TiXI [2] provides an Application Programming Interface (API) for access to CPACS data. As CPACS is based on XML-technology with some additions, these specifics are hidden behind one interface. TiXI is available for several programming languages such as C, C++, Python, Java and Fortran.

**TiGL.** TiGL [3] provides an Application Programming Interface (API) for higher functions regarding CPACS data. TiGL is the geometry library related to CPACS, based on open-source CAD-kernel OpenCASCADE [4]. It provides access to geometry information and several basic calculation algorithms.

### 3.3 Simulation Workflow System - RCE

The simulation framework *RCE* [5] (Remote Component Environment) is an open-source product from the DLR institute Simulation and Software Technology [6], department Distributed Systems and Component Software and the Fraunhofer Institute for Algorithms and Scientific Computing (SCAI). *RCE* is a workflow component framework for distributed computing based on the Eclipse Rich Client Platform (RCP) and provides core functions needed in a distributed environment [7, 8]. It executes a distributed workflow which looks and acts like a data flow diagram. These simulation workflows consist of several components which represent different simulation models. These simulation models can be system and data independent. That means each component wraps the specific simulation code with its special data or operating system dependencies and provides an interface to the simulation framework *RCE*. This has several advantages:

- The Simulation model is independent from other models. The interfaces between them are *RCE* internal
- Possibility to mix different technologies (e.g., programming languages)
- Simulation models are network-compatible regardless of their technology
- Other actors cannot see implementation details of a simulation model
- Programmer can define a fixed interface to their simulation model
  - Reduce possibility of wrong usage
  - Possibility to change interface of the simulation model without interfering *RCE* interface

In the workflow editor area (Figure 2), rectangles represent the simulation models and edges represent the data flow. In general it is possible to create and execute simulation workflows with these two elements. On the right side different available components can be chosen to build more complex workflows, e.g., Python component or other simulation models provided at DLR. The lower part of the screenshot shows three different viewparts. On the lower left side all data available from a simulation workflow can be displayed, analyzed or compered with previous workflow-runs. The lower middle shows the console output of all simulation models involved in a simulation workflow. In the lower right corner a graphical representation of the geometry information in CPACS is presented. Nevertheless there are more functionalities in *RCE* (e.g., options for pre- and post processing of data or execute parametric studies or doing optimization).

[2] http://tixi.googlecode.com, 01.03.2013
[3] http://tigl.googlecode.com, 01.03.2013
[4] http://www.opencascade.org, 01.03.2013
[5] http://www.rcenvironment.de, 01.03.2013
[6] http://www.dlr.de/sc, 01.03.2013

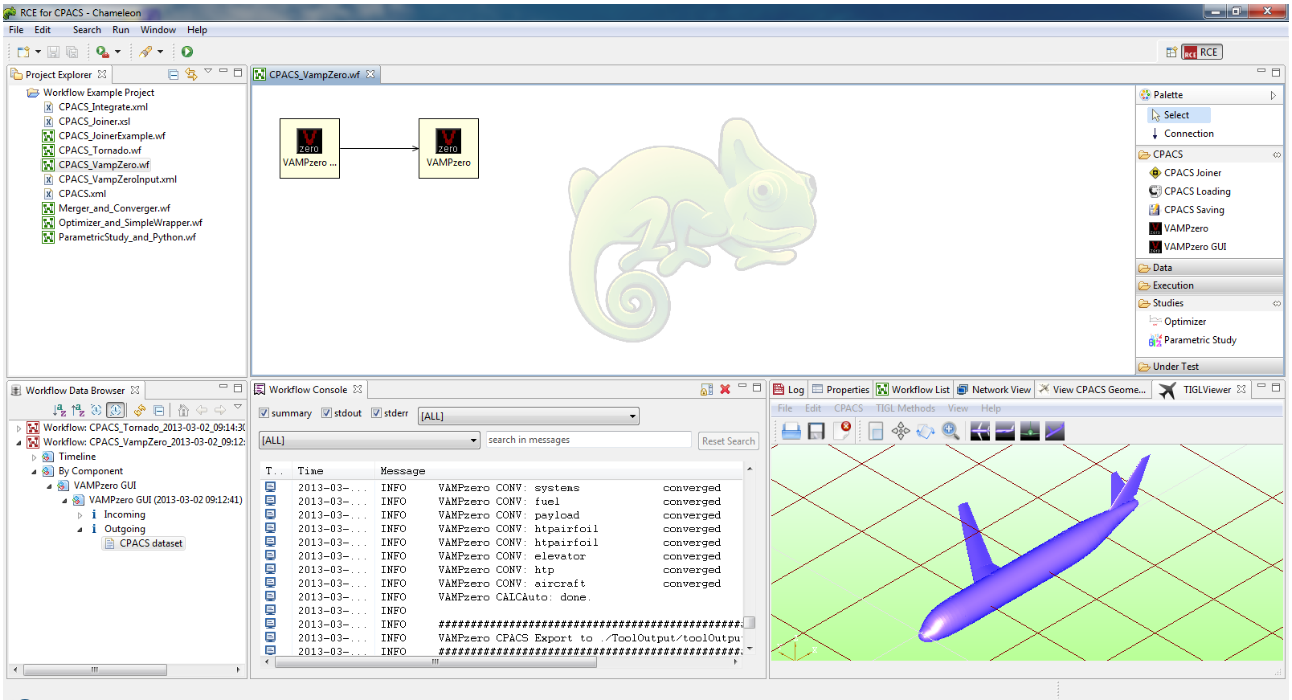


Figure 2: Workflow editor view of *RCE for CPACS*.

### 3.3.1 RCE for CPACS

RCE for CPACS is a special distribution of RCE regarding the CPACS data format. It has several additions for a more common way to work with CPACS in a workflow and simulation model environment. From a workflow systems point of view there are two simulation aspects which should be taken care of. The integration of a simulation model into the workflow system is necessary with respect to its behavior and its interface to CPACS. Simulation data should be integrated into the workflow system as well.

**ToolWrapping.** A wrapping approach was developed and evolved many times (see [8], [9], [10] and current documentation [7]). It connects the simulation model with a special CPACS-related concept to the simulation workflow system. This concept has a contractual manner and ensures that both parts have enough understanding of each other. The concept includes an abstraction layer. All simulation aspects (workflow, model and data) are related to each other but there is a well defined interface which ensures that small changes do not affect other parts.

**XPathChooser.** The main tasks of the engineer need that he is able to address one specific data item in the complete CPACS data format. The XPathChooser dialogue window let him choose the correct node in CPACS for further processing. It provides a special hierarchical view on CPACS data with its specific approach to point to one data item. This graphical element is used at several points in RCE for CPACS, every time the user should have access to a specific data item in CPACS.

**TiGLViewer.** TiGL as a geometry library has an on-top build viewer for CPACS geometry. This viewer component is integrated into RCE for CPACS to have a look onto the current CPACS geometry during execution of a simulation workflow (Figure 2, right lower corner). It is often used for quick error recognition as a specialized view on data is much more intuitive and easier to use than a textual or hierarchical representation.

**XML Editor and Compare.** CPACS is based of XML. Therefore it is obvious to add a specific XML editor to RCE for CPACS. It can be used to show or edit CPACS data. There is

[7] www.rcenvironment.de, 01.03.2013

also a comparing view available for comparing two CPACS datasets, e. g., at different times or locations in a workflow.

## 4 Next Steps on the Road to MDAO

The road to a success-story of third generation MDAO systems is already stepped on even the finish-line is not overstepped yet. This also concerns *RCE for CPACS* as well as other simulation systems. The goal of a useful and accepted MDAO system can be achieved with two different approaches. The first one is to sketch how a possible final software should work and look like and how it should be developed. The second approach is to evolve a software with involving aviation pre-design engineers until the software reaches the goal. We chose the second approach as there is a strong learning curve for aircraft engineers and MDAO software vendors. Also there is no accordance in the community how a third generation MDAO system should look like and therefore no complete concept-description is available. In our opinion it is neccessary to learn from each other and fullfilling little steps towards a final system. This is the reason we want to share our good-practice rules and discuss it.

To come to an outlook of our next steps two aspects should be considered. The first one is understanding MDAO in the context of a distributed simulation including different persons and knowledge domains and how they want to work together. The second aspect is to enhance *RCE for CPACS* as a technology carrier with experience gained in understanding MDAO as a collaborative and interdisciplinary process.

But how we want to move forward in the next months?

- Collecting more good-practice rules via analyzing current available software regarding similarities and distinctions

- Analyzing current working process with *RCE for CPACS* with respect to what performs well and what problems and drawbacks we should face

- Sharing experience with users, researchers and other aviation pre-design engineers

- Enhancing *RCE for CPACS*
  - Bind dataset CPACS in a more intuitive and integrated way in graphical user interface
  - Performing usability studies with respect to pre-design work-tasks
  - Bind pre-design work-tasks in a more intuitive and integrated way in graphical user interface
  - Focus common dataset more into center of simulation (checking validity, consistency)

In our opinion a consistent and trustable dataset is one of the key features of an intelligent MDAO system. The second key feature is to face every work-task of every involved person and to integrate it into the software. The challenge is hereby to identify the important tasks from an avation pre-design view, implement a task-supporting approach in the software and to map it to the mental model of the each involved engineer. This becomes even more important and complex when different approaches of performing pre-design will collaborate.

## 5 Conclusion

Current third generation MDAO systems do not consider the relationship between the simulation workflow, its underlying simulation models and the data processed in the simulation. There are several drawbacks this situation brings into effect. Some effects are related to the quality trustness of the simulation outcome while others affect the control concept of the simulation. If the relationship between these aspects can be considered in a simulation environment many quality-decreasing aspects can be reduced. In addition to the three simulation aspects "workflow", "models" and "data" a fourth aspect "supporting software libraries" is introduced. Our answer to the question how a third generation MDAO system should be designed is to consider the relationships and peculiarities of the workflow system, the simulation models and the data format in every part of the whole simulation environment.

The German Aerospace Center (DLR) developed a third generation MDAO system with consideration of the relationship of all four simulation aspects. It is based on the CPACS data format which holds process and product information, a concept for integrating CPACS into simulation models and a concept to combine CPACS and simulation models into a simulation workflow system called "RCE for CPACS". This solution is a collaborative work of most aeronautic institutes of DLR. The key roles maintain within the Institute for Aerodynamics and Flow Technology, Institute of Air Transportation Systems and Simulation and Software Technology. DLR pursues the objective to evolve this solution to a standard simulation environment for us and all our partners in research and industry. It is still under development and will enhance in the future with ideas from inside DLR and from the growing open source community. Especially collaboration with partners and the integration of their solutions into our software environment is part of our interests.

## References


[1] I. Kroo
Multi-Disciplinary Design Architectures: History and Status. *Multidisciplinary Design Consortium, Stanford University*, 2006.

[2] D. Boehnke et al.
Challenges for Collaborative Data Management in an MDAO Process. *DLRK 2012. German Aeronautics and Space Congress*, 10.-12. September 2012.

[3] D. Boehnke et al.
Towards a Collaborative and Integrated Set of Open Tools for Aircraft Design *ASM 2013. 51st AIAA Aerospace Sciences Meeting*, January 7.-10. 2013.

[4] B. Nagel et al.
Communication in Aircraft Design: Can We Establish a Common Language? *ICAS 2012. 28th International Congress of the Aeronautical Sciences*, September 23.-28. 2012.

[5] D. Boehnke et al.
Evaluation of Modeling Languages for Preliminary Airplane Design in Multidisciplinary Design Environments. *DLRK 2012. German Aeronautics and Space Congress*, 31. August - 2. September 2010.

[6] C. Liersch, M. Hepperle
A Distributed Toolbox for Multidisciplinary Preliminary Aircraft Design. *CEAS 2011. CEAS Aeronautical Journal p. 57-68*, 2011.

[7] D. Seider et al.
Open Source Software Framework for Applications in Aeronautics and Space. *IEEE 2012. Aerospace Conference*, March 2012.

[8] A. Bachmann, M. Litz, M. Kunde, and A. Schreiber
Advances in Generalization and Decoupling of Software Parts in a Scientific Simulation Workflow System. *ADVCOMP 2010. Fourth International Conference on Advanced Engineering Computing and Applications in Sciences*, October 25.-30. 2010.

[9] A. Bachmann, M. Litz, M. Kunde, and A. Schreiber
A dynamic data integration approach to build scientific workflow systems. *IWWM 2009. In: 4th International Conference on Grid and Pervasive Computing: International workshop on workflow management*, May 04.-08. 2009.

[10] A. Bachmann, M. Litz, M. Kunde, Lothar Bertsch and A. Schreiber
Automation of Aircraft Pre-design Using a Versatile Data Transfer and Storage Format in a Distributed Computing Environment. *ADVCOMP 2009. Third International Conference on Advanced Engineering Computing and Applications in Sciences*, October 11.-16. 2009.